Universitat Politècnica de Catalunya

Barcelona Supercomputing Center

Facultat d'Informàtica de Barcelona

Grau en Enginyeria Informàtica

Disseny i desenvolupament d'un compilador i la seva interfície per a entorns HPC en modelització basada en agents (ABM)

Autor

Xavier Lopez Reynau

Ponent

Dr. Josep Casanovas

Directora

Carla Diví Cuesta

27 de juny de 2019

Agraïments

Aquest projecte s'ha beneficiat enormement del suport de moltes persones, algunes de les quals m'agradaria sincerament agrair. Vull agrair al meu ponent Josep Casanovas pels seus útils suggeriments i comentaris constructius durant totes les fases d'aquest projecte. També vull agrair a la meva directora Carla Diví Cuesta per la seva orientació en aquest projecte i les discussions portades a terme sobre el tema. En tercer lloc, vull donar les gràcies als meus amics i companys de la Facultat d'Informàtica de Barcelona. Gràcies a tots vosaltres que sempre heu estat al meu costat durant aquesta experiència. Finalment, però primer en el meu cor, m'agradaria agrair als meus pares pel seu continu suport moral i alè al llarg dels meus estudis.

Resum

Al món de les ciències socials, cada vegada més, se sent la necessitat de portar a terme grans simulacions de diferents fenòmens socials per trobar solucions a diferents problemes, com poden ser moviments migratoris, comportaments socials massius, transmissió de malalties a causa de la interacció entre individus, etc.

Existeixen eines que permeten als professionals de les ciències socials realitzar simulacions d'aquests tipus, però aquestes eines no són prou potents per portar a terme simulacions prou grans per obtenir resultats precisos i realistes. En canvi, existeixen eines molt més potents, però requereixen una sèrie de coneixements tècnics avançats que aquests professionals no tenen i, per tant, no poden utilitzar-les.

Aquest projecte té l'objectiu d'eliminar aquesta barrera d'entrada dels professionals d'àmbits no tècnics a eines de simulació més potents i que els permetin obtenir bons resultats per portar a terme les seves investigacions de forma més precisa i realista.

Resumen

En el mundo de las ciencias sociales, cada vez más, se siente la necesidad de llevar a cabo grandes simulaciones de diferentes fenómenos sociales para encontrar soluciones a diferentes problemas, como pueden ser movimientos migratorios, comportamientos sociales masivos, transmisión de enfermedades debido a la interacción entre individuos, etc.

Existen herramientas que permiten a los profesionales de las ciencias sociales realizar simulaciones de este tipo, pero estas herramientas no son suficientemente potentes para llevar a cabo simulaciones suficientemente grandes para obtener resultados precisos y realistas. En cambio, existen herramientas mucho más potentes, pero requieren una serie de conocimientos técnicos avanzados que estos profesionales no tienen y, por tanto, no pueden utilizarlas.

Este proyecto tiene el objetivo de eliminar esta barrera de entrada de los profesionales de ámbitos no técnicos en herramientas de simulación más potentes y que les permitan obtener buenos resultados para llevar a cabo sus investigaciones de forma más precisa y realista.

Abstract

In the world of social sciences, more and more, there is the need to carry out large simulations of different social phenomena to find solutions to different problems, such as migratory movements, massive social behaviors, transmission of diseases due to interaction between individuals, etc.

There are tools that allow social science professionals to perform such simulations, but these tools are not powerful enough to carry out simulations that are large enough to obtain accurate and realistic results. On the other hand, there are much more powerful tools, but they require a series of advanced technical knowledge that these professionals do not have and, therefore, they can not use them.

This project aims to eliminate this barrier of entry of professionals from non-technical areas to more powerful simulation tools and allow them to obtain good results in order to carry out their research more precisely and realistically.

Glossari

- **Frontend** Part de l'eina de traducció que s'encarrega de la presentació de l'eina de traducció.
- **Backend** Part de l'eina de traducció que conté tota la lògica de la traducció de models.
- **Parser** Analitzador. És el primer component de la lògica del traductor. S'encarrega d'analitzar el codi d'entrada al traductor.
- **Processor** Processador. És el segon component de la lògica del traductor. S'encarrega de generar dades sobre la sortida del primer component del traductor.
- **Generator** Generador. És l'últim component de la lògica del traductor. S'encarrega de generar el codi de sortida del traductor.
- **Map** Mapa de C++. És una estructura de dades que relaciona claus amb valors.

Índex

Capítol 1

Introducció

La modelització basada en agents (ABM, de l'anglès Agent-Based Modeling)[1] és un tipus de modelització computacional utilitzat per simular accions i interaccions entre agents autònoms en un mateix entorn, amb l'objectiu d'avaluar els seus efectes en el sistema en conjunt. És una nova metodologia d'investigació que permet tractar de forma senzilla la complexitat de molts fenòmens no lineals. Tanmateix, aquestes simulacions requereixen alts costos computacionals per ser efectives, i és difícil crear models basats en agents capaços de ser executats en qualsevol mena d'ordinador.

Encara que la simulació ABM és una eina molt utilitzada en ciències tècniques com la física o la biologia, actualment és reconeguda com una de les tècniques amb més potencial per fer simulacions capaces de ser utilitzades en ciències socials, ja que permet explicar com sorgeixen i quins resultats poden donar diferents estructures socials a partir de definir relacions i interaccions entre individus. A més, això ha permès la col·laboració entre ciències socials i ciències tècniques i el sorgiment de nous reptes tecnològics multidisciplinaris que ajuden a portar a terme aquesta col·laboració. Alguns exemples d'aquestes col·laboracions són, per exemple, els articles *On agent-based modeling and computational social science* [23], l'article *Agent-based modeling in social science, history, and philosophy* [24], *Social scientists, qualitative data, and agent-based modeling* [25] i *The impact of agent-based models in the social sciences after 15years of incursions* [26].

Alguns exemples de simulacions que es duen a terme en àmbits socials són les simulacions de corrents migratoris, en les que els agents actuen com a individus migratoris, i les simulacions de propagacions de malalties en diferents comunitats, en les que els agents actuen com a individus que interactuen dins d'aquestes.

1.1 Formulació del problema

Les eines ABM actuals tenen dos inconvenients principals:

1. O bé són eines molt potents però que requereixen coneixements tècnics informàtics elevats per definir els seus models,

2. O bé no són eines escalables, és a dir, no són eines amb prou poder computacional perquè es puguin portar a terme simulacions amb un gran nombre d'agents, molt important a l'hora d'obtenir resultats més precisos i propers a la realitat.

L'eina més utilitzada en l'actualitat per portar a terme simulacions ABM és el software lliure NetLogo [9], desenvolupat inicialment amb l'objectiu de ser utilitzat com a eina d'aprenentatge. Aquesta eina utilitza el llenguatge Logo [6] per definir els seus models, llenguatge també desenvolupat inicialment per ser utilitzat com a eina didàctica per ensenyar programació. Aquesta combinació d'elements el fa una eina molt atractiva per aquelles persones interessades en el desenvolupament de models de simulació ABM, però que no tenen coneixements tècnics informàtics suficients per a desenvolupar models en sistemes més complexos. Aquesta combinació, com s'ha comentat abans, també comporta un problema: NetLogo no ha estat dissenyat per portar a terme simulacions amb un gran nombre d'agents.

En canvi, Pandora [10], una eina de simulació ABM per entorns d'alta computació (HPC) desenvolupada al BSC, és una eina amb prou potencial per a poder executar simulacions amb un gran nombre d'agents, permetent obtenir resultats molt més precisos i realistes, però té l'inconvenient que els models han de ser desenvolupats en C++ o Python. Això comporta un gran desavantatge pels usuaris d'eines ABM d'àmbit de ciències socials, ja que requereix l'aprenentatge del llenguatge de programació C++ per desenvolupar un model per aquesta nova eina. Això fa que molts d'aquests usuaris prefereixin utilitzar eines insuficients a causa de la dificultat d'aprenentatge requerit per l'ús d'eines molt més potents.

1.2 Estat de l'art

Existeixen múltiples eines de simulació ABM en l'actualitat. Moltes d'aquestes han quedat obsoletes a causa de la falta de manteniment i desenvolupament. D'altres, com NetLogo, han guanyat molta popularitat i han esdevingut les eines més utilitzades dins del món de les ciències socials. Per analitzar-les cal fixar-se, principalment, en si és una eina de software

lliure, en quin llenguatge de programació es defineixen els seus models, i si suporta la paral·lelització d'aquests. Un altre aspecte molt important és la visualització de les dades. Per aquesta anàlisi s'ha considerat que un software està obsolet si no presenta cap eina de visualització de resultats, sigui en temps d'execució o a posteriori.

Com ja s'ha comentat anteriorment, NetLogo és una eina de modelatge programable multiagent utilitzada per desenes de milers d'estudiants, professors i investigadors d'arreu del món. És una eina gratuïta, de software lliure desenvolupada sota la llicència GPL (GNU General Public License). Va ser creada l'any 1999 per Uri Wilensky al Center for Connected Learning and Computer-Based Modeling, i posteriorment a Tufts University, Boston. NetLogo està escrit principalment en Scala, amb algunes parts en Java. Els models de NetLogo s'escriuen en el seu propi llenguatge de programació, que és un dialecte del llenguatge de programació Logo. NetLogo no suporta paral·lelització en els seus models, encara que et permet paral·lelitzar múltiples execucions d'un mateix model mitjançant una eina anomenada BehaviorSpace. És una eina ideal per portar a terme petites simulacions en l'àmbit acadèmic, però insuficient per a portar a terme grans simulacions de qualsevol àmbit.

Una altra eina ABM amb certa popularitat a l'àmbit de les ciències socials és MASON [8]. Aquesta és una llibreria software lliure de simulació multi agent d'esdeveniments discrets escrita en Java, dissenyada per ser la base de grans simulacions Java de propòsit personalitzat. Va ser creada l'any 2003 a George Mason University's Evolutionary Computation Laboratory en conjunt amb el GMU Center for Social Complexity. MASON ha estat dissenyat amb l'objectiu de crear una llibreria simple i ràpida per fer simulació ABM, encara que no suporta paral·lelització. Els models s'escriuen en llenguatge Java. MASON és una eina prou potent per a portar a terme simulacions amb un nombre d'agents elevat en ordinadors comuns, encara que tingui la limitació de no poder utilitzar tot el potencial dels processadors gràcies a la paral·lelització. El principal problema d'aquesta eina és que és necessari aprendre Java per desenvolupar els models, inconvenient pels usuaris d'àmbits amb pocs coneixements tècnics. La manca de paral·lelització fa que aquesta eina no pugui ser utilitzada en determinats casos en els quals es necessita un nombre d'agents molt elevat.

Finalment, l'eina que s'està desenvolupant al Barcelona Supercomputing Center (BSC), Pandora, és una llibreria software lliure dissenyada per crear, executar i analitzar simulacions ABM en entorns de computació d'alt rendiment (HPC, de l'anglès High Performance Computing). Els models s'escriuen en llenguatge C++ o Python. També incorpora una

interfície gràfica d'usuari, anomenada Cassandra, que és utilitzada per analitzar els resultats generats per una simulació a Pandora. Aquesta eina és prou potent per a portar a terme grans simulacions no només a ordinadors convencionals, sinó que també és capaç de portar a terme simulacions en entorns HPC per incrementar encara més el nombre d'agents i la complexitat del problema, i reduir el temps d'execució de qualsevol simulació ABM. El principal inconvenient de Pandora és el mateix que el de MASON: el desenvolupament de models requereix coneixements tècnics informàtics, inconvenient per usuaris d'àmbits no tècnics.

En la taula següent es pot veure la comparativa efectuada entre les diferents eines ABM actuals:

Eina ABM	Llicència	Apte per usuaris no tècnics	Paral·lelització
NetLogo	Software Lliure	Si	No
MASON	Software Lliure	No	No
Pandora	Software Lliure	No	Si

Taula 1.1: Comparativa entre eines ABM

L'objectiu de l'equip de desenvolupament de Pandora és aconseguir desenvolupar una eina ABM de software lliure, apte per usuaris no tècnics i que permeti la paral·lelització dels models per poder portar a terme simulacions amb un gran nombre d'agents.

1.3 Objectius del projecte

L'objectiu d'aquest projecte és facilitar la utilització de simuladors ABM de grans prestacions (HPC) a usuaris de l'àmbit de les ciències socials sense coneixements específics de computació.

Per a assolir aquest objectiu, es dissenyarà i desenvoluparà una eina que permeti traduir automàticament un model ABM NetLogo a un codi utilitzable per l'entorn ABM Pandora, desenvolupat pel BSC. Específicament, es volen assolir els següents objectius:

1. Analitzar i comprendre el desenvolupament de models en NetLogo.

2. Dissenyar i implementar una eina de parsing que transformi un codi NetLogo en una estructura de dades AST (Abstract Syntax Tree).

3. Dissenyar i desenvolupar un generador de codi que transformi un model NetLogo a codi C++ per tal d'utilitzar en qualsevol entorn de simulació o d'execució. Això

permetrà, a usuaris amb coneixements mínims de computació, adaptar fàcilment aquest codi a un model apte per ser executat a la plataforma de simulació ABM Pandora.

4. Construir un flux d'execució de models NetLogo a la plataforma Pandora mitjançant una eina de traducció directa entre models d'aquestes plataformes i amb la possibilitat de connectar tot el sistema a l'entorn d'execució del MareNostrum 4, reduint encara més els coneixements en computació necessaris per poder transformar un model NetLogo a un model Pandora.

5. Compatibilitzar tot el procés amb la possibilitat de visualitzar el resultat de la simulació del model ABM generat amb el traductor, mitjançant l'eina de visualització Cassandra, també desenvolupada al BSC.

1.4 Definició de l'abast

1.4.1 Abast

Pel bon compliment dels objectius definits del projecte caldrà fixar quins seran els passos a seguir per dissenyar i desenvolupar tant el compilador com la seva interfície.

El primer pas a realitzar consistirà a estudiar el disseny i funcionament d'un compilador: de quines parts es forma i com aquestes parts interactuen entre elles. Un cop estudiat el disseny i funcionament d'un compilador, s'haurà de fer una recerca per tal de trobar possibles llibreries per facilitar el seu desenvolupament. S'haurà de valorar quina llibreria és la més adequada entre totes les possibles que es trobin, tenint en compte els requisits inicials del projecte i els seus objectius finals. Aquest pas serà molt important, ja que ens permet avaluar el cost del projecte i el temps necessari per a desenvolupar-lo.

Un cop decidida la llibreria a utilitzar, es dissenya i desenvolupa el compilador, mantenint un sistema de tests per assegurar l'eficàcia de la solució. A continuació, es fa un estudi de l'eficiència de la solució i s'apliquen tècniques d'optimització per arreglar els possibles problemes d'eficiència que es puguin trobar.

Un cop verificat el funcionament del compilador, es passa al disseny i desenvolupament de la interfície d'usuari. Per tal d'assegurar la usabilitat de la interfície, es fan diverses proves a usuaris d'àmbits no tècnics i es fa un estudi de les possibles mancances del disseny implementat. S'itera sobre aquest pas fins que el resultat és prou satisfactori.

1.4.2 Possibles obstacles

A l'hora de dissenyar i desenvolupar un projecte d'aquestes característiques cal tenir en compte quins obstacles es poden trobar. Seguint els passos definits prèviament, els possibles errors que es poden trobar són els següents:

- **Errors de disseny**. A l'hora de dissenyar un compilador, és molt important que el disseny a més de tenir en compte l'eficàcia de la solució, també tingui en compte l'eficiència d'aquesta, ja que el temps de còmput pot créixer ràpidament. Per això es farà una cerca i investigació sobre els diferents dissenys de compiladors més utilitzats i els avantatges i inconvenients de cadascun d'ells, per fer una selecció efectiva del disseny a implementar.

- **Errors de programació**. Els errors de programació són els obstacles més comuns que es troben a l'hora de desenvolupar software. Per tal de reduir el seu impacte, es faran validacions de resultats de forma continuada durant tot el desenvolupament del projecte.

- **Calendari**. Es crearà un calendari en el qual es definirà quant de temps es creu que es necessita per desenvolupar cada tasca. Encara així, existeix la possibilitat que per esdeveniments que no podem preveure el calendari s'acabi endarrerint. Per tal de reduir l'impacte que podria generar un d'aquests esdeveniments, es calcularan els terminis a l'alça, de manera que per cada tasca es doni un temps prudencial per trobar solució a qualsevol d'aquests esdeveniments.

1.5 Metodologia

A causa del curt termini d'entrega del projecte i a la facilitat existent en projectes de software de trobar errors organitzatius i d'execució, s'ha decidit que el projecte es desenvoluparà mitjançant la metodologia àgil Scrum, un procés de gestió que redueix la complexitat en el desenvolupament de productes per satisfer les necessitats dels clients mitjançant curtes iteracions i revisions periòdiques [3].

Es faran iteracions setmanals per revisar les tasques descrites i l'avanç del projecte amb els responsables, facilitant la comunicació entre el desenvolupador i els responsables per detectar ràpidament qualsevol mena d'inconvenient que pugui aparèixer. Com que només hi haurà un desenvolupador a l'eina de traducció, no es faran les reunions diàries o "stand-ups". Cada iteració se centrarà en el desenvolupament d'una tasca o d'un conjunt

de tasques concretes amb un objectiu comú, i en la validació d'aquests objectius, tant a curt com a llarg termini.

Al final de cada iteració es farà una avaluació de les tasques desenvolupades, comprovant que els objectius establerts per aquesta s'hagin complert i quins problemes s'han trobat en el seu desenvolupament. Posteriorment, es definiran les tasques a desenvolupar a la següent iteració i se'n farà la seva planificació segons les seves dependències.

Per portar a terme aquesta metodologia s'utilitzarà l'eina anomenada Trello, que permet organitzar el projecte de manera molt simplificada i seguint la metodologia àgil escollida, Scrum. També s'utilitzarà Git com a eina de control de versions.

Capítol 2

Planificació del projecte

2.1 Planificació general

La durada aproximada del projecte és de set mesos. S'inicia el dia 26 de novembre del 2018 i es preveu la seva finalització el dia 28 de juny del 2019, pocs dies abans de l'inici de les presentacions, incloent-hi la preparació del document final i les presentacions de prova prèvies.

2.2 Recursos necessaris

Pel correcte desenvolupament del projecte es necessiten quatre tipus de recursos diferents:

- **Humans**. Es requereix d'un equip humà per tal de desenvolupar el projecte. Aquest equip consta d'un cap de projecte per la part organitzativa, de selecció d'objectius i de validació, i un programador per al disseny de la solució i el seu desenvolupament.
- **Hardware**. Es requereix equipament hardware per tal de desenvolupar el projecte. Aquest equipament consta d'un ordinador per desenvolupar les tasques de disseny i programació del projecte. S'utilitzarà un ordinador portàtil pel baix consum i la flexibilitat que aporta.
- **Software**. Es requereix software d'edició de codi i eines organitzatives. S'utilitzarà Visual Studio Code per l'edició de codi, Trello com a eina organitzativa, i Git com a eina de control de versions. S'han escollit aquestes eines ja que son eines gratuïtes de codi obert.
- **Altres recursos materials**. Es requereix un espai de treball (despatx) amb connexió a Internet i mobiliari bàsic. S'utilitzarà un despatx del BSC compartit amb

un altre equip.

2.3 Descripció de tasques

2.3.1 Planificar el projecte

La planificació del projecte és una de les tasques més importants pel seu correcte desenvolupament. Es defineixen quines són les tasques a desenvolupar, el temps i els recursos que consumiran, així com la seva seqüència lògica i les dependències entre aquestes. També s'especifiquen alternatives a eventuals desviacions, i com aquestes afectarien la duració total del projecte i al consum de recursos.

Aquesta tasca tindrà una durada d'un mes, que és el temps que dura l'assignatura de Gestió de Projectes (GEP). Pel desenvolupament d'aquesta tasca es requerirà l'accés a un ordinador amb connexió a Internet per l'accés als documents formatius i per l'entrega dels documents creats.

2.3.2 Obtenir fonaments bàsics de compiladors

L'obtenció de coneixements bàsics de compiladors és essencial pel correcte desenvolupament de les pròximes etapes de desenvolupament tècnic del projecte. Tenir fonaments bàsics de compiladors és necessari per poder prendre decisions tècniques respecte al disseny i implementació del projecte, pel que es dedicarà una quantitat considerable de temps a l'obtenció d'aquests coneixements.

Aquesta tasca es desenvoluparà en paral·lel a la tasca de planificació, i tindrà una durada de diversos mesos. Requerirà un ordinador amb connexió a Internet per accedir a documentació en línia, a més de documentació impresa accessible a la biblioteca.

2.3.3 Trobar les eines adients pel desenvolupament del projecte

En aquesta etapa es decidirà quines eines informàtiques s'utilitzaran per al desenvolupament del projecte, tenint en compte els objectius finals d'aquest i les característiques de cada una d'aquestes eines. Per decidir si una eina és adequada o no pel projecte, s'haurà de respondre a les següents preguntes:

- Es tracta d'una eina de software lliure?
- Pot ser integrada en un projecte desenvolupat en C++?
- Té un manteniment constant i, per tant, està actualitzada?

- Hi ha una comunitat de desenvolupadors al darrere?

Una vegada escollida l'eina, es requerirà l'estudi i enteniment del funcionament d'aquesta eina per poder desenvolupar la tasca de disseny amb èxit.

Aquesta tasca tindrà una durada d'una setmana, i requerirà un ordinador amb connexió a Internet per fer la cerca, comparació i estudi de les eines de treball disponibles.

2.3.4 Dissenyar el producte

En aquesta etapa es farà el disseny del producte. Per a dissenyar correctament el projecte s'han de tenir en compte els objectius generals d'aquest i l'eina triada en la tasca anterior, ja que aquesta pot requerir seguir un tipus de disseny en particular, sobretot quan es tracta de llibreries per facilitar el desenvolupament del traductor. També s'haurà de dissenyar la interfície d'usuari de manera que sigui senzilla i atractiva pels usuaris d'àmbit no tècnic. Una vegada fet el disseny inicial, aquest es validarà amb un altre membre de l'equip per assegurar que tots els objectius han sigut inclosos en la solució.

Aquesta tasca tindrà una durada de tres dies i requerirà recursos humans per fer la validació d'objectius del disseny desenvolupat.

2.3.5 Desenvolupar un producte viable mínim

En aquesta tasca es desenvoluparà un producte viable mínim (MVP, de l'anglès Minimum Viable Product). Aquest MVP comptarà amb funcionalitats limitades però suficients per poder fer traduccions de models senzills, a més d'una versió inicial de la interfície d'usuari. Es crearan bateries de tests per comprovar el correcte funcionament del producte durant tot el seu desenvolupament. Durant el desenvolupament del MVP, un membre de l'equip validarà que es compleixin els objectius previstos per aquesta tasca setmanalment.

Aquesta tasca tindrà una durada de dos mesos, i requerirà un ordinador amb connexió a Internet per a desenvolupar el projecte, i recursos humans per fer la validació d'objectius del MVP.

2.3.6 Desenvolupar el producte final

En aquesta tasca es finalitzarà el desenvolupament del producte. Aquest producte serà capaç de traduir models basats en agents complexos. Es desenvoluparà una bateria de tests automatitzats i es faran proves d'usabilitat de la interfície amb usuaris externs al projecte i d'àmbit professional no tècnic. Durant el seu desenvolupament, un membre de l'equip validarà que es compleixin els objectius previstos per aquesta tasca setmanalment.

Aquesta tasca tindrà una durada de dos mesos, i requerirà un ordinador amb connexió a Internet per fer el desenvolupament del projecte i recursos humans per fer la validació d'objectius del producte final.

2.3.7 Escriure documentació

En aquesta tasca es finalitzarà la documentació del projecte amb informació més detallada sobre aspectes del projecte que al moment de fer GEP no es van poder descriure. Una vegada completada la documentació, un membre de l'equip la validarà per assegurar que el contingut i la redacció és correcte.

Aquesta tasca tindrà una durada de dues setmanes i requerirà un ordinador amb connexió a Internet per utilitzar les eines per escriure la documentació i recursos humans per fer la validació del document.

2.3.8 Preparar presentació

En aquesta tasca es prepararà la presentació del projecte. Per al correcte desenvolupament d'aquesta tasca, a més de preparar el material de suport per la presentació es faran proves de presentació als membres de l'equip perquè la validin.

Aquesta tasca tindrà una durada d'una setmana i requerirà un ordinador amb connexió a Internet i recursos humans per fer la validació de la presentació.

2.4 Temps estimat

Tasca	**Temps estimat (hores)**
Etapa inicial	
Planificar el projecte	92
Obtenir fonaments bàsics de compiladors	180
Trobar les eines adients pel desenvolupament del projecte	20
Etapa intermitja	
Dissenyar el producte	12
Desenvolupar un producte viable mínim	184
Desenvolupar el producte final	164
Etapa final	
Escriure documentació	40
Preparar presentació	20
Total	712

Taula 2.1: Temps estimat per tasques

2.5 Diagrama de Gantt

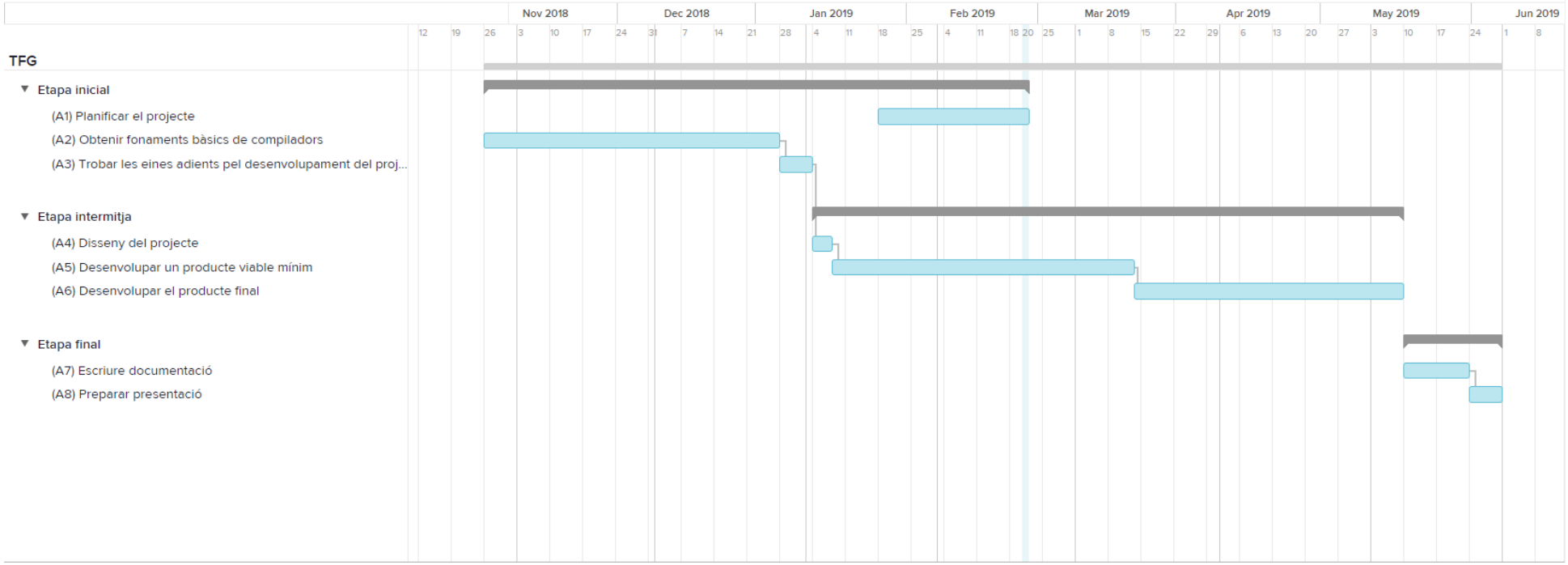

Figura 2.1: Diagrama de Gantt del projecte[2].

2.6 Alternatives i pla d'acció

Durant el desenvolupament del projecte pot haver-hi esdeveniments que provoquin petites desviacions temporals del calendari. Aquests poden ser tan petits problemes de programació com esdeveniments que no permetin avançar en el projecte durant un curt període de temps, com vagues o faltes per malaltia. Per reduir l'impacte d'aquestes, el temps de desenvolupament de les tasques s'ha incrementat per donar marge a aquests possibles petits problemes que es trobin. A més, gràcies a l'ús de la metodologia àgil Scrum, gran part d'aquestes desviacions es poden detectar a temps i es pot reduir el seu impacte de manera que no afecti la planificació general del projecte.

Però també pot haver-hi esdeveniments que provoquin grans desviacions temporals del calendari, com per exemple fenòmens naturals que afectin la infraestructura o mobiliari del projecte, avaries de la infraestructura de treball, faltes per malaltia de llarga durada, errors greus en la planificació del projecte o problemes greus de programació. En cas que es doni un d'aquests esdeveniments, s'haurà de reduir el temps de desenvolupament d'una tasca posterior. Tal com s'ha descrit anteriorment, gran part del temps de desenvolupament del projecte es basa en la programació d'aquest. Per aquest motiu s'ha dividit l'etapa de programació en dues tasques: Desenvolupament del producte viable mínim, i desenvolupament del producte final. Això permetrà, en cas d'una desviació temporal important, reduir temps de la tasca de desenvolupament del producte final per ajustar el calendari a aquests nous desviaments i centrar-se primer en els aspectes fonamentals del projecte.

Capítol 3

Pressupost i sostenibilitat

3.1 Gestió Econòmica

Un cop planificat el projecte, es realitza un estudi econòmic per determinar la viabilitat del projecte a desenvolupar, on s'identifica i estima quins costos té el desenvolupament del projecte, i posteriorment es descriu com es fa el control de costos del projecte.

3.1.1 Identificació i estimació dels costos

Per a la correcta identificació i estimació dels costos del projecte, es divideixen els costos en costos directes, costos indirectes, costos per contingència i costos per imprevists.

A continuació es fa una estimació del cost de personal (salari brut, seguretat social i altres costos de personal) dels diferents rols necessaris per desenvolupar el projecte.

Rol	**Cost anual (€)**	**Cost per hora (€)**
Cap de projecte	46250	21.09
Programador	33750	11.16

Taula 3.1: Costos anuals i per hora de recursos humans

Costos directes

L'estimació dels costos directes es fa a nivell d'activitats del diagrama de Gantt mostrat anteriorment. Es calcularà el cost de l'aportació de cada rol present en el desenvolupament de la tasca segons la seva durada prevista i el salari estimat prèviament.

Activitat	**Rol**	**Hores estimades**	**Cost (€)**
A1	Cap de projecte	92	1940.28
A2	Programador	180	2008.8
A3	Programador	20	223.2
A4	Programador	8	89.28
	Cap de projecte	4	84.36
A5	Programador	156	1740.96
	Cap de projecte	28	590.52
A6	Programador	140	1562.4
	Cap de projecte	24	506.16
A7	Cap de projecte	40	843.6
A8	Cap de projecte	20	421.8
Total		**712**	**10011.36**

Taula 3.2: Costos directes

Costos indirectes

- **Lloc de treball**. Es disposa d'un despatx amb tot l'equipament necessari per portar a terme els projectes de l'equip. També es disposa de servei de neteja i manteniment, a més d'altres serveix com Internet, llum, calefacció i aire condicionat. El cost total del lloc de treball és de 3000€/any, per tant, tenint en compte la durada del projecte (7 mesos) i que som quatre membres al equip, el cost equivalent del projecte es de 437.5€.

- **Hardware**. Es dedicarà un únic ordinador pel desenvolupament total del projecte amb un cost de 1000€. Per reduir els costos i l'impacte ambiental, s'utilitza hardware utilitzat prèviament per altres equips i projectes. Suposant que el temps de vida d'un ordinador portàtil és de 5 anys, el cost de hardware equivalent a la durada del projecte és de 116,66€.

- **Software**. Pel desenvolupament del projecte no es requereix cap software propietari o privatiu, pel que el cost d'aquest serà 0.

- **Transport**. En aquest projecte l'equip de desenvolupament i el client estan a la mateixa empresa, pel que el cost en transport del projecte és 0.

Recurs	**Cost total (€)**	**Cost projecte (€)**
Lloc de treball	3000	437.5
Hardware	1000	116.66
Software	0	0
Transport	0	0
Total		**554.16**

Taula 3.3: Costos indirectes

Contingència

A causa de la llarga durada del projecte i en vista dels costos calculats anteriorment, es decideix reservar una part del pressupost per la partida de contingència.

Concepte	**Cost total (€)**	**Percentatge a aplicar**	**Cost del marge (€)**
Costos directes	10011.36	15%	1501.70
Costos indirectes	554.16	15%	83.12
Total			**1584.82**

Taula 3.4: Costos de contingència

Imprevists

Tal com s'ha especificat anteriorment, pot haver-hi diferents tipus d'imprevists durant el desenvolupament del projecte. Els imprevists de tipus humà (baixes, errors de programació) no constitueixen cap cost addicional per al projecte, ja que gràcies a l'ús de metodologies àgils i a l'organització del projecte en diferents etapes, aquests imprevists queden mitigats fent ús dels plans alternatius descrits. Els imprevists de tipus no humà (avaries) sí que constitueixen un cost addicional pel projecte i han de ser comptabilitzats. En aquest cas s'aprofitaria un ordinador ja utilitzat prèviament per un altre equip, pel que valorarem el cost del canvi d'ordinador en un 5% del seu cost original.

Concepte	**Cost total (€)**	**Percentatge a aplicar**	**Cost del marge (€)**
Hardware	1000	5%	50
Total			**50**

Taula 3.5: Costos imprevists

Cost final

Concepte	**Cost total (€)**
Costos directes	10011.36
Costos indirectes	554.16
Contingència	1584.82
Imprevists	50
Total	**12200.34**

Taula 3.6: Cost final previst del projecte

3.1.2 Control de gestió

Per tal de controlar el cost del projecte, al final de cada fase del projecte es farà una revisió dels seus costos totals. Aquests costos es compararan amb estimacions prèvies per obtenir indicadors de desviació que ens permetran veure la desviació total del pressupost inicial. S'utilitzaran els indicadors següents:

- **Desviació en cost** = (CE - CR) x CHR
- **Desviació en consum** = (CHE - CHR) x CE

On:

- **CE** = Cost Estimat
- **CR** = Cost Real
- **CHR** = Consum d'Hores Real
- **CHE** = Cost d'Hores Estimat

Gràcies al fet que es farà una revisió dels costos de cada tasca en finalitzar-la, podrem determinar, si hi ha una desviació, en quina tasca ha sigut i de quin tipus de desviació es tracta.

Si les desviacions de pressupost són degudes als imprevists mencionats anteriorment, s'utilitzarà la partida d'imprevists per cobrir aquests costos. En cas que no sigui suficient o que les desviacions provinguin d'altres fonts, s'utilitzarà el fons de contingència per a cobrir-les.

3.1.3 Viabilitat econòmica

Aquest projecte, juntament amb Pandora, forma part d'un conjunt de projectes que tenen com a objectiu apropar la comunitat científica al món del High Performance Computing mitjançant la plataforma que prové el BSC, MareNostrum 4. El valor que el BSC obté d'aquesta promoció és molt alt, el que fa que doni finançament per portar a terme tots aquests projectes degut a l'alta viabilitat econòmica que presenten. Per tant, el cost de desenvolupament del projecte és assumit per l'empresa gràcies al gran valor que n'obté a partir d'aquesta promoció.

3.2 Sostenibilitat

3.2.1 Projecte Posat en Producció

El Projecte Posat en Producció (PPP) avalua el projecte durant la seva planificació, desenvolupament i implantació.

Ambiental

Ja que el disseny i desenvolupament d'aquest traductor no requereix cap hardware especial, s'ha decidit utilitzar un ordinador portàtil pel desenvolupament d'aquest projecte. D'aquesta manera es redueix l'impacte ambiental del projecte al mínim. S'han utilitzat dos elements bàsics per analitzar-lo:

- **Hardware**. És sabut que el cost ambiental de producció d'un ordinador és alt [4] [5]. Per aquest motiu s'ha decidit reutilitzar tot el hardware necessari, aprofitant-lo d'altres equips i projectes que ja no el necessiten.
- **Energia**. És sabut que el cost energètic d'un ordinador portàtil és inferior al cost energètic d'un ordinador no portàtil [11]. Per aquest motiu, s'ha decidit utilitzar un ordinador portàtil pel desenvolupament del traductor.

Econòmic

El cost total del projecte, tal com s'ha indicat anteriorment, és de 12,200.34€ i s'ha calculat mitjançant els costos directes, els costos indirectes, els costos de contingència i els costos imprevists. Dins dels costos directes, es poden trobar els costos de recursos materials i de recursos humans desglossats per les diferents etapes descrites al diagrama de Gantt. El cost total del projecte es pot veure desglossat en la taula 3.6.

Social

A nivell personal i d'equip aquest projecte aportarà coneixements en el desenvolupament de compiladors i gramàtiques, alhora que aporta coneixements addicionals en el desenvolupament d'eines de modelització ABM. Per tant, podem considerar aquest projecte com una font de coneixements tècnics.

3.2.2 Vida Útil

La vida útil avalua el projecte un cop desenvolupat i durant la seva execució. No es preveu el seu desmantellament, ja que és complement d'un altre projecte més gran dins del BSC i no se'n preveu el seu tancament o desmantellament.

Ambiental

Com que es tracta de software que s'executa de forma esporàdica i és apte de ser executat en qualsevol ordinador, el cost ambiental directe d'aquest és mínim.

Però, com s'ha comentat anteriorment, aquest projecte pretén apropar els entorns HPC a usuaris d'àmbits no tècnics, de manera que s'està incrementant el nombre d'usuaris final d'aquests entorns, els quals consumeixen grans quantitats d'energia. En concret, el supercomputador MareNostrum 4, que és el supercomputador més gran d'Espanya i pel que s'està desenvolupant aquesta eina, té un consum energètic de 1.3 MWatt a l'any [7]. Encara que no sigui un cost ambiental directe del projecte, s'ha considerat un aspecte important i a tenir en compte a l'hora de reflexionar sobre el cost ambiental d'aquesta eina, i es busca utilitzar mesures que permetin reduir aquest cost ambiental.

Econòmic

Com que és un projecte que resol un problema molt específic i d'un producte propietat també del BSC, no podem trobar altres productes similars per fer-ne comparacions. Encara així, podem veure de forma general com aquesta eina millorarà i afegirà valor a l'eina a la qual complementa, Pandora, que seguirà sent completament gratuïta, i millorarà el seu potencial i la seva utilitat respecte a la resta d'eines ABM existents.

El manteniment del projecte tindrà un cost molt inferior al cost del desenvolupament del PPP, ja que només requerirà manteniment en el cas en què es modifiqui qualsevol dels dos llenguatges de desenvolupament de models, tant de NetLogo com de Pandora. Com que és un esdeveniment poc probable, podríem dir que el cost de manteniment serà negligible.

Social

El desenvolupament d'aquest projecte permetrà a nous usuaris d'àmbits no tècnics aproparse a eines més tècniques i potents amb un esforç molt menor al que poden necessitar per utilitzar altres eines de simulació ABM de potència més o menys similar. Com a conseqüència, això permet realitzar simulacions molt més realistes i detallades d'aspectes socials com poden ser les migracions o propagacions de malalties, i que poden donar resultats amb un valor social molt més elevat que amb eines menys tècniques i potents.

Per tant, incrementa la capacitat dels usuaris d'obtenir resultats d'àmbits socials que milloren notablement aspectes molt diversos de la societat i, per tant, la vida de les persones.

3.3 Identificació de lleis i regulacions

Com que aquest software no emmagatzema ni utilitza cap tipus de dades d'usuaris, no hi ha cap llei o normativa vigent de protecció de dades que se li pugui aplicar. A més, les llibreries de les que se serveix són llibreries de codi obert que permeten l'ús comercial i no comercial, pel que tampoc hi ha risc d'infringir cap llicència de codi.

Capítol 4

Disseny

L'eina de traducció de models forma part d'un projecte molt més gran desenvolupat per l'equip HPC modelling and simulation for Societal Challenges (HPC4SC) del BSC. Aquest projecte está format per diferents eines:

- Pandora, una eina de simulació ABM per entorns d'alta computació (HPC) amb possibilitat de portar a terme simulacions al MareNostrum 4.
- Cassandra, una eina de visualització dels resultats obtinguts a les simulacions de Pandora.
- L'eina de traducció de models que s'està desenvolupant en aquest projecte final de grau.

Totes aquestes eines treballen de manera conjunta per assolir l'objectiu final de l'equip: Facilitar la utilització de simuladors ABM de grans prestacions (HPC) a usuaris de l'àmbit de les ciències socials sense coneixements específics de computació.

El flux de treball, visualitzat gràficament a la figura 4.1, constaria d'una primera part de traducció del model NetLogo. Aquesta traducció podría fer-se de dues maneres diferents:

- Traducció del model NetLogo directament al llenguatge de programació C++. Això permetria a un usuari amb coneixements de C++ aplicar la traducció a qualsevol entorn de simulació, inclòs Pandora.
- Traducció del model NetLogo a un model Pandora. D'aquesta manera, qualsevol usuari podrà utilitzar la plataforma Pandora i, per tant, entorns d'alta computació mitjançant aquesta eina.

Una vegada traduït el model, aquest es compilarà juntament amb la plataforma de simulació ABM Pandora, per generar un arxiu binari que podrà ser executat al MareNostrum 4. Aquesta execució generarà un output, el qual es podrà utilitzar en la plataforma de visualització de dades Cassandra, també desenvolupada al BSC.

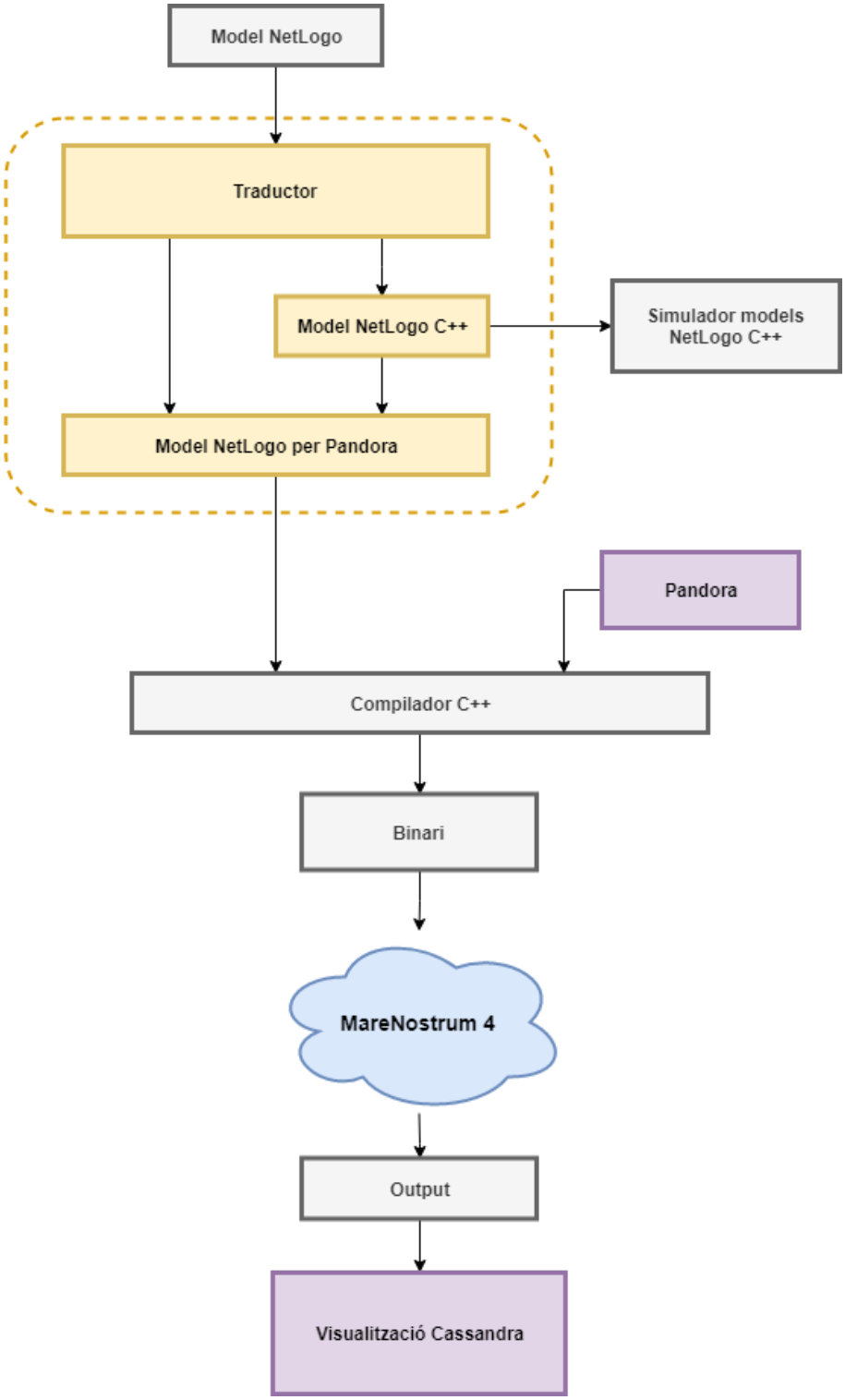

Figura 4.1: Diagrama del projecte

4.1 Backend

El backend és l'encarregat de traduir un model NetLogo directament a codi C++ o, per altra part, a un model ABM compatible amb Pandora, escrit també en llenguatge de programació C++. El procés de traducció de models és el següent:

1. El model NetLogo és utilitzat com a input del primer component del flux de treball del backend, el parser. Aquest, amb l'ajuda de les gramàtiques desenvolupades, transformarà el model NetLogo en una estructura de dades interna representada com un AST (Abstract Syntax Tree).

2. A continuació, l'AST és utilitzat com a input del segon component del backend, el processor. Aquest component és l'encarregat de portar a terme la inferència de tipus i altres procediments per tal de facilitar la feina del següent component.

3. Finalment, tant l'AST com les dades generades pels processors són utilitzades com a input pel tercer component del backend, el generator. Aquest component serà l'encarregat de generar la traducció, sigui directament a C++ o a un model compatible amb Pandora.

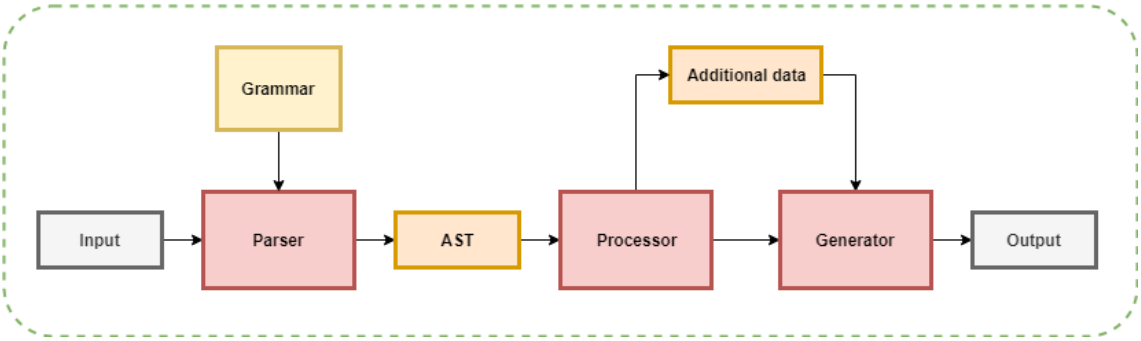

Figura 4.2: Diagrama del backend

4.1.1 Parser

El parser és l'encarregat de transformar el model NetLogo en una estructura de dades anomenada Abstract Syntax Tree (AST)[13].
Un AST és una representació en forma d'arbre de l'estructura sintàctica d'un codi font escrit en un llenguatge de programació. Cada node de l'arbre representa un element o construcció existent en el codi font. És molt utilitzat en el desenvolupament de compiladors i sovint s'utilitza com a representació intermèdia del programa durant les diferents etapes del procés de compilació.

La transformació del codi font a l'AST es fa mitjançant gramàtiques que utilitzen un format similar al Extended Backus Naur Form (EBNF)[12], les quals identifiquen diferents aspectes sintàctics d'un model NetLogo i els transformen en diferents nodes de l'arbre sintàctic, com per exemple, expressions aritmètiques o condicionals.

4.1.2 Processor

El processor és l'encarregat de processar, classificar i obtenir informació de l'AST per facilitar el desenvolupament de la següent etapa del procés de traducció. Com que un model de NetLogo difereix en molts aspectes d'un model de Pandora, l'etapa de processing en la traducció de models NetLogo a models Pandora agafa molta força i es fa vital pel correcte funcionament del traductor, ja que serveix de pont entre aquestes dues tecnologies. El processor està format per diferents components que tenen funcionalitats diferents, que són els següents:

- Inferència de tipus: S'encarrega de detectar de forma automatitzada el tipus d'una expressió dins del codi font[21]. A causa de la diferència de tipat entre els models NetLogo i C++ (tipat dinàmic i tipat estàtic, respectivament) es fa completament necessari implementar aquest component per fer possible la traducció.

- Cerca d'accions: S'encarrega de buscar quines accions han d'executar els agents dins del codi font del model NetLogo. Pandora requereix la definició d'Accions pel desenvolupament dels seus models. En canvi, NetLogo utilitza unes expressions i estructures pròpies per executar funcions sobre els seus agents. Per aquest motiu, és necessària una cerca prèvia per trobar aquestes accions, per posteriorment poder generar les classes corresponents per Pandora.

- Cerca de funcions auxiliars: La definició de contextos a NetLogo i Pandora difereix en el fet que pel primer és implícit, i pel segon és explícit. Per tant, s'ha de fer una cerca i recorregut pel flux d'execució per detectar quines crides es fan des de els diferents contexts que puguin existir, per tal de poder definir correctament aquests a Pandora.

4.1.3 Generator

El generator és l'encarregat de transformar l'AST en codi C++ o un model C++ compatible amb la plataforma Pandora. Aquest procediment es du a terme mitjançant l'AST i la informació obtinguda a partir dels diferents components del processor.

En cas de la traducció directa entre models NetLogo i models Pandora, el generator s'encarregarà de generar les següents classes C++:

- World: És l'encarregat de portar a terme l'execució de tota la simulació del model. Representa l'espai en què es durà a terme la simulació, i és l'encarregat de la inicialització i execució dels rasters (diferents espais en els quals un agent pot posicionar-se i executar accions) i els agents.
- Agent: Representa un agent de la simulació. Aquest conté una sèrie d'atributs i accions a executar durant la simulació. Pot interactuar amb altres agents i amb el món mitjançant les accions assignades.
- Action: Representa una acció que pot ser portada a terme per un Agent en un pas de simulació. Permet la interacció entre l'agent que l'executa i la resta d'agents i el món.

4.2 Frontend

El frontend és l'encarregat de connectar el backend amb l'usuari, fent més senzilla la interacció amb l'eina de traducció de models. Aquesta eina, per tal de facilitar el seu ús i eliminar la barrera d'aprenentatge tecnològic darrere del traductor, s'ha dissenyat de forma minimalista i amb el mínim d'interaccions possible.

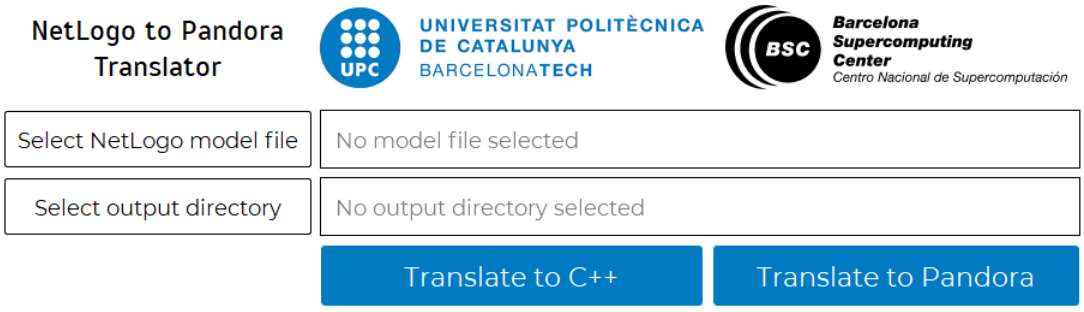

Figura 4.3: Disseny de la interfície

Capítol 5

Implementació

En aquest capítol es parlarà de les decisions preses durant la implementació del projecte i com aquestes han afectat la planificació i desenvolupament general del projecte.
La implementació del projecte s'ha dividit en quatre components, corresponents a les diferents fases del backend i al component frontend: parser, processor, generator i interface.

5.1 Parser

La implementació del parser s'ha fet principalment utilitzant la llibreria Spirit.Qi de Boost, que permet escriure gramàtiques i definicions formals utilitzant un format similar al EBNF directament en C++. Els motius pels quals s'ha decidit utilitzar aquesta llibreria són els següents:

- Simplicitat: Permet escriure gramàtiques en format EBNF directament en C++, de manera que s'eviten passos intermedis de compilació per passar del EBNF a codi C++. Com a conseqüència, s'evita tenir eines addicionals per compilar, preprocessar o integrar els diferents components del projecte. En resum, simplifica el procés de desenvolupament del projecte.

- Correctesa: Boost és un conjunt de llibreries àmpliament provades i utilitzades per multitud de projectes que fins i tot algunes d'aquestes han estat incloses a l'estàndard de C++[15][19]. Això dóna la seguretat que aquestes llibreries són correctes i, per tant, segures d'utilitzar en un producte comercial.

- Llicència: La llicència de Boost[16] fomenta l'ús comercial i no comercial de les seves llibreries sense cap cost addicional, pel que es converteix en una de les millors opcions per portar a terme el projecte.

Altres llibreries que s'han plantejat i probat son Guidelines Support Library (GSL)[18], BDE Development Environment (BDE)[14], The Parsing Expression Grammar Template Library (PEGTL)[20] i C++11 header-only Parsing Expression Grammars (PEG)[17], però no han estat utilitzades ja que no han semblat tan convenients com el conjunt de llibreries Boost pel desenvolupament d'aquest component.

La implementació de les gramàtiques ha suposat gran part de l'etapa de desenvolupament del projecte, ja que tal com es comentarà més endavant al capítol de Reptes, l'estructura sintàctica de NetLogo té certes complicacions que han fet d'aquesta etapa una de les més complexes del projecte.

```
variable_declaration = lexeme[lit("let") >> " "] > var >> -(expr);

assignment = lexeme[lit("set") >> " "] > var > expr;

if_statement =
        lexeme[lit("if") >> " "] >> expr >> '[' > +statement_ > ']' > -( "else" > +statement_ )
    |   lexeme[lit("ifelse") >> " "] >> expr >> '[' > +statement_ > ']' > '[' > +statement_ > ']';


while_statement = lexeme[lit("while") >> " "] > '[' > expr > ']' > '[' > +statement_ > ']';

ask_agent = lexeme[lit("ask") >> " "] >> agent > expr > '[' > +statement_ > ']';
ask_agentset = lexeme[lit("ask") >> " "] >> agentset_expression_ > '[' > +statement_ > ']';

create_agentset = "create-" > agentset > expr > -('[' > +statement_ > ']');

move_statement = direction >> expr;

setxy_statement = lexeme[lit("setxy") >> " "] >> expr >> expr;
```

Figura 5.1: Fragment de codi corresponent a una de les gramàtiques del Parser

5.2 Processor

La implementació dels diferents components que formen el processor s'ha fet sense utilitzar cap llibreria addicional a les ja esmentades amb anterioritat. Utilitza alguns components de Boost per poder recórrer amb facilitat l'AST generat en la fase anterior i genera diferents estructures de dades per facilitar l'etapa posterior del traductor.

Les estructures de dades generades per cadascun dels components del processor són les següents:

- Inferència de tipus (TypeInference.cpp)
  - Tipus de funcions (function_types): Estructura de dades que guarda el tipus de retorn de cada funció del model. Ha estat implementat amb un map de

l'estàndard de C++, en el que la clau és el nom de la funció i el contingut és el tipus de retorn.

  - Tipus dels arguments de funció (function_args_types): Estructura de dades que guarda el tipus de cadascun dels arguments que accepta una funció. Com que NetLogo defineix els seus models amb tipat dinàmic, no és possible esbrinar el tipus dels arguments d'una funció només amb la seva definició, pel que s'ha d'inferir pel context i les crides efectuades a la funció. Ha estat implementat amb un map de l'estàndard de C++, en el que la clau es el nom de la funció i el contingut és una llista de tipus que representa els tipus dels arguments ordenats.
  - Tipus de variables (variable_types): Estructura de dades que guarda el tipus de les variables definides al codi font. Ha estat implementat de la mateixa manera que l'estructura function_types.
  - Tipus de variables globals (global_variable_types): Estructura de dades que guarda el tipus de les variables globals definides al codi font. Ha estat implementat de la mateixa manera que function_types i variable_types.

- Cerca d'accions (AgentActions.cpp)

  - Accions d'agents (agent_actions): Estructura de dades que guarda les diferents accions d'agents trobades. La finalitat d'aquesta estructura de dades és facilitar la generació de la classe Agent i les diferents classes Action. Ha estat implementat amb un map de l'estàndard de C++, en el que la clau és un nom generat per aquesta acció i el contingut és el node de l'AST corresponent a aquesta acció.
  - Funcions auxiliars d'agents (agent_aux_functions): Estructura de dades que guarda una llista de noms de funció que un agent requereix per portar a terme les seves accions. Com que en NetLogo una mateixa funció pot ser utilitzada per diferents contexts, cal trobar quines funcions auxiliars requereix cada tipus d'agent diferent.

- Identificació de la inicialització (ScanSetupFunction.cpp)

  - Funció de inicialització d'agents (agentset_setup): Estructura de dades que guarda el node corresponent a la inicialització dels agents del model. La finalitat d'aquesta estructura de dades es facilitar la cerca d'accions auxiliars al processor explicat anteriorment.

– Funcions auxiliars d'inicialització (setup_aux_functions): Estructura de dades que guarda una llista de noms de funció que es requereixen per fer la inicialització del model. La finalitat d'aquesta estructura és facilitar la generació de tota la part d'inicialització del model.

5.3 Generator

La implementació del generator s'ha fet sense utilitzar llibreries addicionals a les ja utilitzades amb anterioritat. Gràcies a l'AST i a les estructures de dades generades pels components del processor, la generació directa de codi C++ s'ha simplificat. Però, degut a les diferències en estructura i execució dels models entre NetLogo i Pandora, la part de generació de models Pandora és d'una major complexitat i s'ha dividit en dues parts:

- Generació de la inicialització: A partir de la funció de setup de NetLogo, s'obté informació sobre la inicialització de la simulació i es genera el codi d'inicialització de les classes del model implicades, com per exemple la classe World, que s'encarrega de la creació dels agents i dels rasters, i la classe Globals, que conté les variables globals definides al model NetLogo.

- Generació del codi de simulació: A partir de la funció go de NetLogo, s'obté informació sobre l'execució de la simulació i es genera el codi d'execució del model de les classes implicades en l'execució. Aquestes classes representen totes les Accions que els agents poden dur a terme, com els mateixos Agents i la generació de la cua d'accions que porten a terme a cada pas de simulació.

5.4 Interface

La implementació de la interfície de l'eina de traducció s'ha fet utilitzant la llibreria de desenvolupament d'interfícies d'usuari Qt[22].
El desenvolupament de la interfície s'ha fet de manera independent a l'eina de traducció, de manera que no hi ha cap tipus d'acoblament entre el frontend i el backend. D'aquesta manera, es pot realitzar el desenvolupament de qualsevol de les dues parts de forma completament independent.
El funcionament d'aquesta interfície és molt senzill. S'encarrega d'enviar la informació proveïda per l'usuari (direcció de l'arxiu del codi font del model NetLogo dins del sistema, i directori on es generarà el codi del model Pandora dins del sistema) a l'eina de traducció.

Una vegada s'ha enviat la informació a l'eina de traducció, aquesta s'encarregarà de traduir el codi i generar els arxius on pertoca.

Capítol 6

Validacions

Durant el desenvolupament del projecte, per assegurar i validar el correcte funcionament de l'eina de traducció, s'han portat a terme dos tipus de validacions durant el projecte:

1. Validacions lògiques: Al llarg de tot el desenvolupament del projecte, s'han fet validacions lògiques de forma contínua per tal de verificar que els resultats obtinguts en cada moment eren correctes. Aquestes validacions s'han fet de manera manual, comparant els resultats obtinguts amb els resultats esperats. Aquestes validacions les he anat efectuant jo a mesura que anava desenvolupant l'eina de traducció.

2. Validació d'objectius: Durant el desenvolupament del projecte, de forma periòdica s'han fet validacions d'objectius per tal d'assegurar que aquests eren clars. Aquestes validacions han estat efectuades per altres membres de l'equip de forma manual.

Gràcies a aquestes validacions, s'ha pogut corroborar els objectius assolits i avançar a les següents fases o etapes del desenvolupament de l'eina de traducció de models ABM.

Capítol 7

Reptes

Durant el desenvolupament d'aquest projecte s'ha trobat una sèrie de reptes tècnics que han suposat més temps del que s'havia previst a la planificació, pel fet que les diferències entre els models de NetLogo i Pandora són més de les que s'havien previst.

7.1 Característiques sintàctiques de NetLogo

El llenguatge de desenvolupament de models de NetLogo té una sèrie de característiques sintàctiques que fan del desenvolupament d'aquesta eina una tasca més complexa de l'habitual:

1. No hi ha cap indicador de final de statement.
2. Les crides de funcions a NetLogo no tenen delimitadors d'arguments.

Aquestes característiques per separat en principi no comporten cap complexitat addicional pel desenvolupament del traductor, però en conjunt provoquen problemes d'ambigüitat complexos. En concret, aquestes dues característiques en conjunt permeten tindre expressions com la següent:

```
a b c d e f g h i
```

Aquesta expressió es pot interpretar de diferents maneres, com per exemple:

- a(b, c) d(e, f) g(h, i)
- a(b, c(d, e), f) g(h, i)
- a b c(d, e(f(g(h(i()))))))

Solucionar aquests problemes d'ambigüitat no es gens senzill. Per això s'han adoptat diferents mesures per reduir aquest nivell d'ambigüitat i reduir la complexitat del parser:

1. Una funció només pot tenir variables com a argument

2. Les funcions han de ser declarades prèviament a la seva utilització (com a C++). Això permet al parser saber quants arguments requereix una funció.

Gràcies a aquestes mesures, s'ha aconseguit reduir prou la complexitat del parser i continuar amb el desenvolupament de l'eina de traducció sense grans desviacions en la planificació.

7.2 Inferència de tipus

NetLogo és un llenguatge de programació de tipat dinàmic. En canvi, C++ és un llenguatge de programació de tipat estàtic. A causa d'aquesta diferència, la generació dels models de Pandora no es pot dur a terme directament sense un pas intermedi entre l'etapa de parsing i l'etapa de generació del model.
Aquesta problemàtica ha comportat la necessitat de desenvolupar un algorisme d'inferència de tipus que ajudés a solucionar aquest problema, però sense afegir grans desviacions a la planificació establerta.
Finalment, es va decidir implementar un algorisme d'inferència de tipus molt simple, en el que seguint unes regles bàsiques pugues inferir amb certesa el tipus de les diferents expressions i funcions que el model requerís pel seu desenvolupament.
Aquest algorisme es basa en una cerca recursiva per l'AST, visitant les expressions i funcions que es requerissin per la generació posterior del model de Pandora. Mitjançant una sèrie de regles molt bàsiques, aquest component és capaç de decidir si continuar cercant el tipus d'una expressió o funció, o si amb la informació actual ja és capaç d'inferir el tipus del node en qüestió.
Lluny de ser un algorisme general d'inferència de tipus, és suficient per als objectius actuals del projecte i fàcilment ampliable amb el creixement de l'eina de traducció.

7.3 Diferències estructurals

NetLogo és una eina de desenvolupament de models de simulació que va ser creada amb objectius acadèmics i que ha anat evolucionant fins a l'eina que és actualment. Això implica una sèrie de diferències estructurals amb els models de Pandora que han provocat desviacions en la planificació inicial, a causa de les complicacions que aquestes diferències comporten en la generació dels models.
Els models de NetLogo estan dividits en dos components principals:

1. Inicialització: S'inicialitzen totes les variables globals i es defineixen tots els elements de la simulació (agents, patchs, links i observer). També es decideix quants d'aquests elements s'executaran durant la simulació.

2. Bucle de simulació: Es defineix una funció que s'executarà a cada pas de simulació i que serà l'encarregada de decidir quins i com s'executaran els diferents elements de la simulació.

En canvi, els models de Pandora estan formats de diferents elements (world, agents i actions, tal com s'ha explicat al capítol Disseny), cadascun d'ells corresponent a una classe C++ diferent, amb la seva pròpia inicialització i execució.
Aquestes diferències fan que la traducció de NetLogo a Pandora no sigui una traducció literal, sinó que s'ha de fer una anàlisi exhaustiu dels models i mitjançant els components adients del processor, generar un codi mitjançant el generator on les inicialitzacions i funcionalitats de simulació siguin al seu lloc adient dins del model de Pandora.

Capítol 8

Integració de coneixements

La realització d'aquest projecte ha sigut possible gràcies als coneixements adquirits en les assignatures següents:

- LP (Llenguatges de Programació): Assignatura obligatòria d'especialitat on vaig aprendre els conceptes bàsics de compiladors i diferents paradigmes de programació que m'han ajudat a planificar i desenvolupar el projecte.
- TC (Teoria de la Computació): Assignatura obligatòria d'especialitat on vaig aprendre els conceptes i la definició d'autòmats, expressions regulars i gramàtiques, sense els quals hagués sigut molt difícil portar a terme aquest projecte.
- LI (Lògica a la Informàtica): Assignatura obligatòria d'especialitat on vaig aprendre els conceptes bàsics de lògiques que m'han ajudat durant el plantejament de les gramàtiques.
- PROP (Projectes de Programació): Assignatura obligatòria on vaig aprendre eines i nocions bàsiques de treball en equip en projectes de desenvolupament de software que han estat presents durant tot el desenvolupament del projecte.

Capítol 9

Conclusions

9.1 Contribucions

Els experts en ciències socials sense coneixements tècnics que volen portar a terme simulacions al seu àmbit es troben amb el problema que, moltes vegades, les eines que tenen a la seva disposició i que saben fer servir no són prou potents.
Aquest projecte aporta una solució a aquesta problemàtica permetent a aquests usuaris accedir a eines més sofisticades sense els coneixements tècnics necessaris per utilitzar-les. Això permet obtenir resultats més precisos i realistes en les solucions que duen a terme i, per tant, que la societat se'n pugui beneficiar dels seus estudis amb més rapidesa i seguretat.
És per això que considero que el meu treball fa una gran contribució tant en l'àmbit tècnic com en l'àmbit social, ja que permet la col·laboració entre aquests dos camps d'estudi de manera que els resultats i beneficis que obté la societat són molt millors.

9.2 Assoliment d'objectius

L'objectiu d'aquest projecte és, tal com s'ha explicat amb anterioritat, dissenyar i desenvolupar una eina que permeti traduir automàticament un model ABM NetLogo a un codi utilitzable per l'entorn ABM Pandora.
Més concretament, s'han definit una sèrie d'objectius específics, dels quals analitzarem el seu assoliment:

- Analitzar i comprendre el desenvolupament de models en NetLogo. Aquest objectiu ha estat assolit, ja que era necessari per al correcte desenvolupament del primer component del traductor, el parser.

- Dissenyar i implementar una eina de parsing que transformi un codi NetLogo en una estructura de dades AST. Aquest objectiu també ha estat assolit en la seva totalitat amb la finalització del primer component del traductor, el parser.

- Dissenyar i desenvolupar un generador de codi que transformi un model NetLogo a codi C++ per tal de ser utilitzat en qualsevol entorn de simulació o execució. Aquest objectiu també ha estat assolit en la seva totalitat gràcies al desenvolupament de la primera versió del generador de codi, que tradueix un model NetLogo directament a C++.

- Construir un flux d'execució de models NetLogo a la plataforma Pandora mitjançant una eina de traducció directa entre models d'aquestes plataformes. Aquest objectiu ha estat assolit en la seva majoria i s'està treballant en la seva finalització i validació.

- Compatibilitzar tot el procés amb la possibilitat de visualitzar el resultat de la simulació del model ABM mitjançant Cassandra. Aquest objectiu ha estat assolit en la seva majoria i s'està esperant la finalització de l'objectiu anterior per passar a la validació d'aquest últim objectiu.

9.3 Valoració personal

Personalment aquest projecte m'ha aportat molts coneixements tècnics a causa del fort component d'autoaprenentatge que hi ha al darrere, però a més m'ha ajudat a conèixer i aprendre a valorar la planificació d'un projecte i com adaptar-se a les seves possibles desviacions per tal de poder arribar a l'objectiu final.
També m'ha servit per prendre consciència de la gran necessitat que tenen els professionals d'àmbits no tècnics de trobar eines útils i potents en àmbits d'alta computació que els permetin utilitzar-les sense haver d'adquirir coneixements tècnics específics per aquestes.

9.4 Treball futur

Com que és un producte que forma part d'un projecte més gran i en continu desenvolupament com es Pandora, el seu desenvolupament continuarà més enllà d'aquest TFG. Per tant, el treball futur d'aquest projecte passa per la continuació del seu desenvolupament i de forma paral·lela al desenvolupament de Pandora, a més de trobar solució a una sèrie de limitacions que s'han trobat durant el desenvolupament d'aquest TFG definides al capítol Reptes d'aquesta memòria.

Bibliografia

[1] Colloquium paperadaptive agents, intelligence, and emergent human organization: Capturing complexity through agent-based modeling: Agent-based modeling: Methods and techniques for simulating human systems. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC128598/. (Accessed on 03/19/2019). 1

[2] Diagrama de gantt - google drive. https://drive.google.com/file/d/15M3eJ1BUzKzGCtYr2_yi2mdelr6E6rKB/view. (Accessed on 03/20/2019). 13

[3] Homepage — scrum.org. https://www.scrum.org/. (Accessed on 03/19/2019). 6

[4] How do computers pollute the environment? — bizfluent. https://bizfluent.com/info-8699749-do-computers-pollute-environment.html. (Accessed on 03/19/2019). 19

[5] How do laptops affect the environment? — sciencing. https://sciencing.com/laptops-affect-environment-23252.html. (Accessed on 03/19/2019). 19

[6] Logo programming language. http://el.media.mit.edu/logo-foundation/what_is_logo/logo_programming.html. (Accessed on 03/19/2019). 2

[7] Marenostrum — bsc-cns. https://www.bsc.es/es/marenostrum/marenostrum. (Accessed on 03/19/2019). 20

[8] Mason multiagent simulation toolkit. https://cs.gmu.edu/~eclab/projects/mason/. (Accessed on 03/19/2019). 3

[9] Netlogo home page. https://ccl.northwestern.edu/netlogo/. (Accessed on 03/19/2019). 2

[10] Pandora: An hpc agent-based modelling framework — bsc-cns. https://www.bsc.es/research-and-development/software-and-apps/software-list/pandora-hpc-agent-based-modelling-framework. (Accessed on 03/19/2019). 2

[11] Review of computer energy consumption and potential savings. https://www.dssw.co.uk/research/computer_energy_consumption.html. (Accessed on 03/11/2019). 19

[12] Extended Backus Naur Form, Sep 1998. [Online; accessed 22. Jun. 2019]. 25

[13] Abstract syntax tree - Wikipedia, Jun 2019. [Online; accessed 22. Jun. 2019]. 24

[14] BDE Standard Library, Jun 2019. [Online; accessed 23. Jun. 2019]. 28

[15] Boost C++ Libraries, Jun 2019. [Online; accessed 23. Jun. 2019]. 27

[16] Boost Software License, Jun 2019. [Online; accessed 23. Jun. 2019]. 27

[17] cpp-peglib, Jun 2019. [Online; accessed 23. Jun. 2019]. 28

[18] GSL: Guidelines Support Library, Jun 2019. [Online; accessed 23. Jun. 2019]. 28

[19] ISO/IEC JTC1/SC22/WG21 - The C++ Standards Committee - ISOCPP, Jun 2019. [Online; accessed 23. Jun. 2019]. 27

[20] taocpp/PEGTL, Jun 2019. [Online; accessed 23. Jun. 2019]. 28

[21] Type inference - Wikipedia, Jun 2019. [Online; accessed 22. Jun. 2019]. 25

[22] The Qt Company. Qt | Cross-platform software development for embedded & desktop, Jun 2019. [Online; accessed 26. Jun. 2019]. 30

[23] Rosaria Conte and Mario Paolucci. On agent-based modeling and computational social science. *Frontiers in Psychology*, 5:5–668, 07 2014. 1

[24] Dominik Klein, Johannes Marx, and Kai Fischbach. Agent-based modeling in social science, history, and philosophy : An introduction. *Historical Social Research / Historische Sozialforschung*, 43:7–27, 04 2018. 1

[25] Roman Seidl. Social scientists, qualitative data, and agent-based modeling. *ETH Zurich*, 2014. 1

[26] Flaminio Squazzoni. The impact of agent-based models in the social sciences after 15 years of incursions,. *History of Economic Ideas*, LVIII, 01 2010. 1